VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018

An Internship Project Report
on

UBER FARE PREDICTION

Submitted in partial fulfilment of the requirements as a part of

Artificial Intelligence and Machine Learning Internship

for the award of a degree of

Bachelor of Engineering in

Information Science and Engineering

Submitted by

SRI PRIYA G

1RN18IS108

Under the Guidance of

Mrs. Vinutha G K

Assistant Professor

Department of ISE

Department of Information Science and Engineering

RNS Institute of Technology

Dr. Vishnuvaradhana Road, Rajarajeshwari Nagar post, Channasandra, Bengaluru-560098

2021-2022

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru - 560098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the Internship work entitled ***Uber Fare Prediction*** has been successfully completed by **SRI PRIYA G(1RN18IS108)** a bonafide student of **RNS Institute of Technology, Bengaluru** in partial fulfilment of the requirements of 8th semester for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year **2021-2022**. The internship report has been approved as it satisfies the academic requirements in respect of internship work for the said degree.

Mrs. Vinutha G K	**Dr. Suresh L**	**Dr. M K Venkatesha**
Internship Guide	Professor and HoD	Principal
Assistant Professor	Department of ISE	RNSIT
Department of ISE	RNSIT	

External Viva

Name of the Examiners	**Signature with Date**
1. _____	**1.** _____
2. _____	**2.** _____

DECLARATION

I, **SRI PRIYA G [USN: 1RN18IS108]** student of VIII Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Internship work entitled ***Uber Fare Prediction*** has been carried out by us and submitted in partial fulfilment of the requirements for the *VIII Semester degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2021-2022.

Place: Bengaluru

Date: 12th January 2022

SRI PRIYA G

(1RN18IS108)

ABSTRACT

Predictive analytics uses archival data to predict the future events. Typically, past data is used to build a mathematical model that captures important trends. That predictive model is then used on current data to predict the future or to suggest actions to take for optimal outcomes. Predictive analytics has received a lot of attention in recent years due to advances in supporting technology, particularly in the areas of big data and machine learning.

Companies also use predictive analytics to create more accurate forecasts, such as forecasting the fare amount for a cab ride in the city. These forecasts enable resource planning for instance, scheduling of various cab rentals to be done more effectively. For a cab rental start-up company, the fare amount is dependent on a lot of factors.

This project aims to understand all patterns and to apply analytics for fare prediction. The proposed work is to design a system that predicts the fare amount for a cab ride in the city. The aim is to build regression models, which will predict the continuous fare amount for each cab ride and help prediction depending on multiple time-based, positional and general factors.

ACKNOWLEDGMENT

At the very onset I would like to place our gratefulness to all those people who helped me in making the Internship a successful one.

Coming up, this internship to be a success was not easy. Apart from the sheer effort, the enlightenment of the very experienced teachers also plays a paramount role because it is they who guided me in the right direction.

First of all, I would like to thank the **Management of RNS Institute of Technology** for providing such a healthy environment for the successful completion of internship work.

In this regard, I express sincere gratitude to our beloved Principal **Dr. M K Venkatesha,** for providing us all the facilities.

We are extremely grateful to our own and beloved Professor and Head of Department of Information science and Engineering, **Dr. Suresh L**, for having accepted to patronize me in the right direction with all her wisdom.

We place our heartfelt thanks to **Mrs. Vinutha G K,** Assistant Professor, Department of Information Science and Engineering for having guided internship and all the staff members of the department of Information Science and Engineering for helping at all times.

I thank **Mr. Aman Upadhyay, NASTECH,** for providing the opportunity to be a part of the Internship program and having guided me to complete the same successfully.

I also thank our internship coordinator **Dr. R Rajkumar,** Associate Professor, Department of Information Science and Engineering. I would thank my friends for having supported me with all their strength and might. Last but not the least, I thank my parents for supporting and encouraging me throughout. I have made an honest effort in this assignment.

SRI PRIYA G

Table of Contents

LIST OF FIGURES

List of Table

Chapter 1

INTRODUCTION

New York City taxi rides paint a vibrant picture of life in the city. The millions of rides taken each month can provide insight into traffic patterns, road blockage, or large-scale events that attract many New Yorkers. With ridesharing apps gaining popularity, it is increasingly important for taxi companies to provide visibility to their estimated fare and ride duration, since the competing apps provide these metrics upfront.

Predicting fare and duration of a ride can help passengers decide when is the optimal time to start their commute, or help drivers decide which of two potential rides will be more profitable, for example. Furthermore, this visibility into fare will attract customers during times when ridesharing services are implementing surge pricing. In order to predict duration and fare, only data which would be available at the beginning of a ride was used. This includes pickup and dropoff coordinates, trip distance, start time, number of passengers, and a rate code detailing whether the standard rate or the airport rate was applied. Linear regression, decision tree regression, and random forest models were used to predict duration and fare amount.

The workings of uber dynamic model plays an important role in the price prediction model to predict the price of the ride form given source to destination we use the data such as the distance between the source and destination. As weather and the holiday or festive season plays a very important role in deciding the surge in the price of the cab. And with the help of machine learning algorithms, we are able to visualize the data into pictures or graphs for better understanding and visualizing the estimation of the pricing with various factors.

The traffic also plays a major role in calculating the surge of price of the ride with increase of the traffic the availability of the cabs becomes limited and when the demand for the cabs start to increase the service provider will not be able to provide the cabs this causes the surge in the price during the peak hours. The peak hours are usually calculated as the time when there is a large number of requests for the cabs and the price is increased during the peak hour. With the help of the driver's data set as well as the customers dataset we are able to calculate the peak hour for each day.

1.1 MACHINE LEARNING

Machine learning is a subset of artificial intelligence in the field of computer science that often uses statistical techniques to give computers the ability to "learn" (i.e., progressively improveperformance on a specific task) with data, without being explicitly programmed.

The name machine learning was coined in 1959 by Arthur Samuel. Evolved from the study ofpattern recognition and computational learning theory in artificial intelligence, machinelearning explores the study and construction of algorithms that can learn from and make predictions on data – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs.Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or infeasible; example applications include email filtering, detection of network intruders or malicious insiders working towards a data breach, optical character recognition (OCR), learning to rank, and computer vision.

Machine learning is closely related to (and often overlaps with) computational statistics, which also focuses on prediction-making through the use of computers. It has strongties to mathematical optimization, which delivers methods, theory and application domains tothe field. Machine learning is sometimes conflated with data mining, where the latter subfieldfocuses more on exploratory data analysis and is known as unsupervised learning. Machine learning can also be unsupervised and be used to learn and establish baseline behavorial profiles for various entities and then used to find meaningful anomalies.

1.2 ARTIFICIAL INTELLIGENCE

Artificial intelligence (AI), sometimes called machine intelligence, is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans and other animals. In computer science AI research is defined as the study of "intelligent agents": any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals. Colloquially, the term "artificial intelligence" is applied whena machine mimics "cognitive" functions that humans associate with other human minds, suchas "learning" and "problem solving". Modern machine capabilities generally classified as AI include successfully understanding human speech, competing at the highest

level in strategic game systems (such as chess and Go), autonomously operating cars, intelligent routing etc.

Artificial intelligence was founded as an academic discipline in 1956, and in the years since has experienced several waves of optimism, followed by disappointment and the loss offunding (known as an "AI winter"), followed by new approaches, success and renewed funding. For most of its history, AI research has been divided into subfields that often fail to communicate with each other. These sub-fields are based on technical considerations, such asparticular goals (e.g. "robotics" or "machine learning"), the use of particular tools ("logic" or artificial neural networks), or deep philosophical differences. Subfields have also been based on social factors (particular institutions or the work of particular researchers).

The traditional problems (or goals) of AI research include reasoning, knowledge representation, planning, learning, natural language processing, perception and the ability to move and manipulate objects. General intelligence is among the field's long-term goals. Approaches include statistical methods, computational intelligence, and traditional symbolic AI. Many tools are used in AI, including versions of search and mathematical optimization, artificial neural networks, and methods based on statistics, probability and economics. The AIfield draws upon computer science, mathematics, psychology, linguistics, philosophy and many others.

The field was founded on the claim that human intelligence "can be so precisely described that a machine can be made to simulate it". This raises philosophical arguments about the nature of the mind and the ethics of creating artificial beings endowed with human-like intelligence which are issues that have been explored by myth, fiction and philosophy since antiquity. Some people also consider AI to be a danger to humanity if it progresses unabated. Others believe that AI, unlike previous technological revolutions, will create a risk of mass unemployment.

In the twenty-first century, AI techniques have experienced a resurgence following concurrent advances in computer power, large amounts of data, and theoretical understanding. AI techniques have become an essential part of the technology industry, helping to solve manychallenging problems in computer science, software engineering and operations research. In simple terms, AI aims to extend and augment the capacity and efficiency of mankind in tasks of remaking nature and governing the society through intelligent machines, with the final goalof realizing a society where people and machines coexist harmoniously together.

1.3 PROBLEM STATEMENT

The project is about the Uber cab company who has done its pilot project and now they are looking to predict the fare for their future transactional cases. And this cab company delivers services to lakhs of customers daily. Now it becomes really important to manage their data properly to come up with new business ideas to get best results. In this case, earn most revenues. So, it becomes really important estimate the fare prices accurately.

Chapter 2

LITERATURE SURVEY

Author J. Chao tells us the popularity of uber in the recent years and about the urban citizens who are benefited by the uber. Later the author compares the difference between the competitive taxis and uber and defines new way of calling and also the new way of paying for cabs, the author also tells us about the importance of data produced by the cabs daily and also about the visualization and analysis of data. After that the author tells how the different time and different environments will have an effect on passengers to make different choices.

A large part of the literature focuses primarily on spatial-temporal analysis and demand & price forecasting of the taxi trips. Popular destinations in New York City were uncovered by detecting popular drop-off locations linear regression model and implemented to analyse the travel patterns in urban areas for effective investment decisions in rapid transit infrastructure and service (Hochmair, 2016) and density analysis was used to detect demand fluctuations (Markou, Rodrigues, & Pereira, 2017). The travel time estimation problem was addressed with a big data-driven approach using a simple baseline for Travel Time Estimation (Wang, Kuo, Kifer, & Li, 2014). With the aid of the Semi-Markov process and approximate dynamic programming (ADP) approach, time of the pricing was determined with supply and demand fluctuations of taxis and resulted in increasing the market revenue by 10% (Qian & Ukkusuri, 2017).

Random Forest Regression model was implemented to build a system that delivers traffic insights and recommendations to help taxi drivers with useful guidelines (Ibrahim & Shafiq, 2019). A taxi searching algorithm with an accuracy of 97.59% accuracy was built using distributed coordination and clustering to minimize the time of taxi reaching the passengers (Agrawal, Raychoudhury, Saxena, & Kshemkalyani, 2018). The problem of predicting the number of taxis required in zones at various times was solved using STVec (smooth transaction vector error correction model) and multi-outputs support vector regression (MSVR) model (Zhou, Wu, Wu, Chen, & Li, 2015). Visual Exploration is a challenge for big Spatio-Temporal urban taxi data, this was overcome by building a model that uses an adaptive level-of-detail rendering strategy to create a visualization that is clutter-free for results that are large (Ferreira, Poco, Vo, Freire, & Silva, 2013). The Markov predictor model built for prediction of taxi demand was 89% accurate and better by 11%

compared to neural network predictors (Zhao, Khryashchev, Freire, Silva, & Vo, 2016). A platform for distributed spatiotemporal analytics was built to deal with the heterogeneous spatiotemporal dataset (Deva, Raschke, Garzon, & Kupper, 2017) and a simulation combined with effective indexing scheme and parallelization was built to get a scalable approach (Ota, Vo, Silva, & Freire, 2015).

As a baseline prediction, the mean duration and fare from the training set were used to predict a constant value for the validation set, Linear Regression Model is used. To avoid selecting a sub-optimal model by selecting covariates by hand, forward selection was used to identify which subset of covariates would be best to use. When iteratively adding the variables that minimizes the RSS one at a time it is evident that selecting a model with all covariates does not improve the Cp score (proxy for test error) over a 20-covariate model. Therefore, for simplicity reasons, the smallest model was selected for linear regression.

As traffic is clustered and aggregated more densely to different locations at different times, the location of the ride will clearly have an affect on the trip duration. Although there is no straightforward way of considering all locations between the start and end points of a ride, the pickup and dropoff locations are available in the dataset and can be used to model some of the effect of traffic and conjunctions. In the linear regressions, the locations' effect on trip duration is modeled simply by the magnitude of the longitude and latitude coordinates. As traffic is clearly not varying solely based on the magnitude of the coordinates, the linear models fail to account for the nonlinear effect the locations have on traffic and hence trip duration (and also fare amount). An algorithm that can better account for these nonlinearities is the random forest. The random forest algorithm aggregates many decision trees built on bootstrapped samples of the training data in order to reduce the high variance of a single decision tree and improve prediction accuracy [7][8]. Each of these decision trees aims to divide the predictor space, i.e; the set of all possible values for the features x1, x2, ..., xn, in J distinct and non-overlapping regions R. The predictor space is 1 R2. RJ divided into high-dimensional rectangles, with the goal to find rectangles R, that minimize the RSS, where y is the mean response for the training observations $\frown$ Rj within the jth rectangle. When building each tree, a top-down approach is taken. Beginning with all points in the same region, the algorithm successively splits the predictor space into two halves, stopping when there are no more than five points in a region. At each split, a prediction x and cutpoint j s are chosen such that splitting the predictor space into the regions

{x | x } and leads to the biggest j < s {x | xj ≥ s} reduction in RSS. Once the regions are defined, a prediction by a single tree is made by averaging the responses of the training observations in the region to which the test observation belongs. In the random forest, a large number of trees are fit, each using a bootstrap sample from the training data, and a prediction of a new observation is made using the mean of the predictions by all the trees. At each split, only m of the total n predictors are randomly chosen to be considered. This approach is taken to decorrelate the trees, as considering all predictors might yield very similar trees when one or a few predictors are particularly strong. As averaging many uncorrelated trees leads to a larger reduction in variance, this approach often yields better prediction results. As can be seen in figure 3, the model performs better for a smaller choice of m . Also, averaging over a larger number of trees yields a better results, although the effect is flattening out after a few hundred trees. To optimize prediction accuracy, m = √n and 500 trees were used.

Chapter 3

ANALYSIS

3.1OBJECTIVE OF THE PROJECT

The study aims to analyse various key aspects of taxi trips that generate more revenue by achieving the following objectives:

1. To classify the trips into various segments based on profitability.
2. To carry out spatiotemporal analysis to assess the net migration of taxi from one location to another at various times of the day to maintain demand and supply equilibrium of cabs.
3. To build a dynamic price prediction model to balance margin and conversion rates.

3.2 SCOPE OF THE PROJECT

The objective of this project is to predict accurate fare amount for the users. Introducing machine learning model for data analysis and the importance of data produced by the cabs on daily basis and how this data can be used by the machine learning to tell the consumer about the exact price of their ride before starting the ride. This provides the consumer to make better choice of cab based on the price predicted by the Machine-Learning model. The proposed system uses the cab dataset to make predictions for each ride booked by the customer.

3.3 HARDWARE REQUIREMENTS

Memory	8 GB or 4GB
Graphics Card	AMD Radeon RX 480 or NVIDIA GeForce GTX 970
CPU	Intel Core i5-4590
File Size	2 GB
OS	Windows 7 SP1

Table 3.1: Hardware Requirement

3.4 SOFTWARE REQUIREMENTS

Application	Google CoLab
Dataset	Uber Prediction Data Set
Language	Python

Table 3.2: Software Requirement

3.5 SOFTWARE DESCRIPTION

Python: Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object- oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Python uses dynamic typing and a combination of reference counting and acycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Python's design offers some support for functional programming in the Lisptradition. It has filter, map and reduce functions; list comprehensions, dictionaries,sets, and generator expressions. The standard library has two modules (itertools andfunctools) that implement functional tools borrowed from Haskell and Standard ML.

Dataset: The aim is to build regression models that will predict the continuous fare amount for each of the cab-rides depending on multiple time-based, positional and generic factors. This problem statement falls under the category of forecasting which deals with predicting continuous values for the future (the continuous value is the fare amount of the cab ride). Figuire.1 shows a sample of the data set that will be used to predict the fare amount of a cab ride.

	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
0	2009-06-15 17:26:21.0000001	4.5	2009-06-15 17:26:21+00:00	-73.844311	40.721319	-73.841610	40.712278	1
1	2010-01-05 16:52:16.0000002	16.9	2010-01-05 16:52:16+00:00	-74.016048	40.711303	-73.979268	40.782004	1
2	2011-08-18 00:35:00.00000049	5.7	2011-08-18 00:35:00+00:00	-73.982738	40.761270	-73.991242	40.750562	2
3	2012-04-21 04:30:42.0000001	7.7	2012-04-21 04:30:42+00:00	-73.987130	40.733143	-73.991567	40.758092	1
4	2010-03-09 07:51:00.000000135	5.3	2010-03-09 07:51:00+00:00	-73.968095	40.768008	-73.956655	40.783762	1

Figure 3.1: Sample Dataset

There are six predictor variables and one target variable which are listed as follows:

Predictors:

1) Pickup_datetime: timestamp value indicating when the cab ride started.

2) Pickup_longitude: float for longitude coordinate of where the cab ride started.

3) Pickup_latitude: float for latitude coordinate of where the cab ride started.

4) Dropoff_longitude: float for longitude coordinate of where the cab ride ended.

5) Dropoff_latitude: float for latitude coordinate of where the cab ride ended.

6) Passenger_count: an integer indicating the number of passengers in the cab ride.

Target: fare_amount

Data structures upon proper data type conversion are shown in Figuire.2:

```
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   key                5000 non-null   object
 1   fare_amount        5000 non-null   float64
 2   pickup_datetime    5000 non-null   object
 3   pickup_longitude   5000 non-null   float64
 4   pickup_latitude    5000 non-null   float64
 5   dropoff_longitude  5000 non-null   float64
 6   dropoff_latitude   5000 non-null   float64
 7   passenger_count    5000 non-null   int64
dtypes: float64(5), int64(1), object(2)
memory usage: 312.6+ KB
```

Here we can see there are 8columns in which 6 numerics and 2 are object.

Figure 3.2: Data structures datatype conversions

3.6 SOFTWARE TOOLS

- **Google Colab**: Google colab is a free and open-source distribution of the Python programminglanguages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify packagemanagement system and deployment. Package version similar to that of Python3 Juypter Notebook.
- **Python libraries**: For the computation and analysis we need certain python libraries which are used to perform analytics. Packages such as sklearn, numpy, pandas, surprise andmatplotlib are needed.
    i) **Sklearn:** It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k- means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

    ii) **NumPy:** NumPy is a general-purpose array-processing package. It provides a high- performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.

    iii) **Pandas:** Pandas is one of the most widely used python libraries in data science. It provides high-performance, easy to use structures and data analysis tools. Unlike NumPy library which provides objects for multi-dimensional arrays, Pandas provides in-memory 2d table object called Data frame.

    iv) **Matplotlib:** Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine, designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib. Matplotlib was originally written by John D. Hunter.

    v) **Seaborn:** Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and color palettes to make statistical plots more attractive. It is built on the top of matplotlib library and also closely integrated to the data structures from pandas. Seaborn aims to make

visualization the central part of exploring and understanding data. It provides dataset-oriented APIs, so that we can switch between different visual representations for the same variables for better understanding of the dataset.

vi) **OS:** The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The *os* and *os.path* modules include many functions to interact with the file system.

vii) **Warnings:** Warning messages are typically issued in situations where it is useful to alert the user of some condition in a program, where that condition (normally) doesn't warrant raising an exception and terminating the program. For example, one might want to issue a warning when a program uses an obsolete module. Warning messages are normally written to sys.stderr, but their disposition can be changed flexibly, from ignoring all warnings to turning them into exceptions. The disposition of warnings can vary based on the warning category, the text of the warning message, and the source location where it is issued. Repetitions of a particular warning for the same source location are typically suppressed.

Chapter 4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

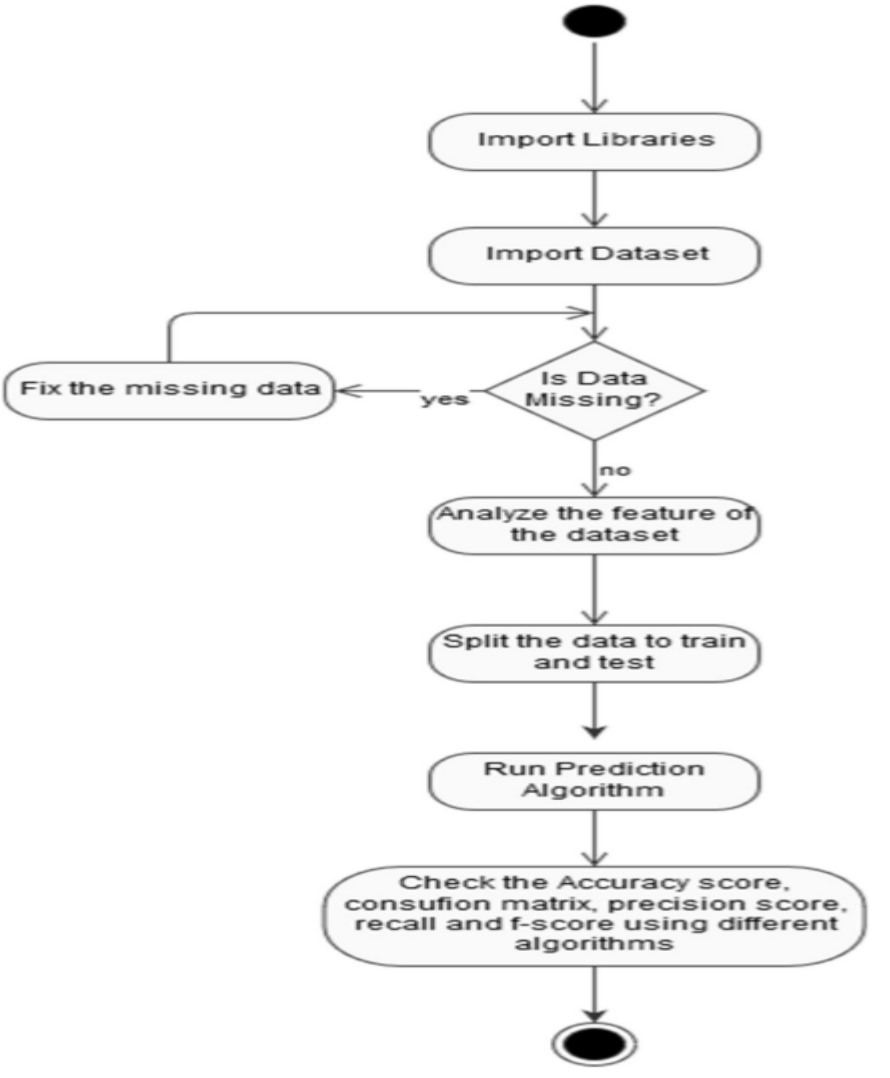

Figure 4.1: Activity Diagram

The above figure is the flow chart of activity done in the project.

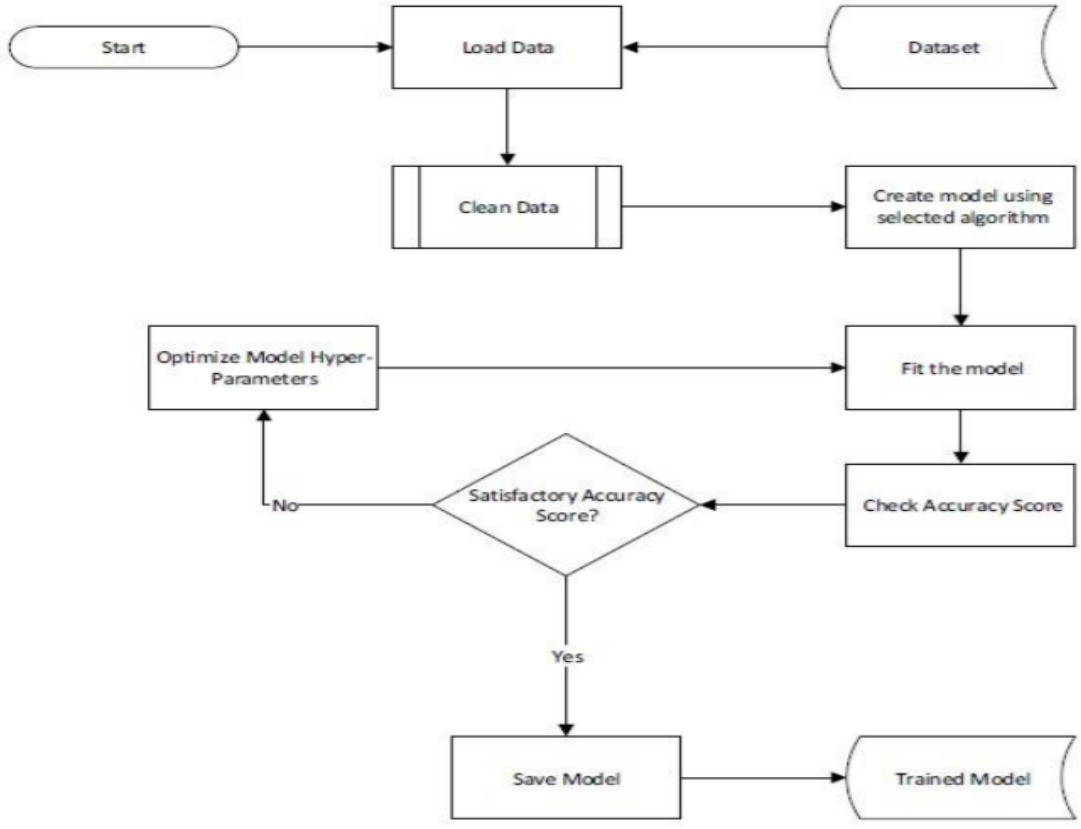

Figure 4.2: Training Model Process

In the above figure is the flow diagram of how the training model takes places.

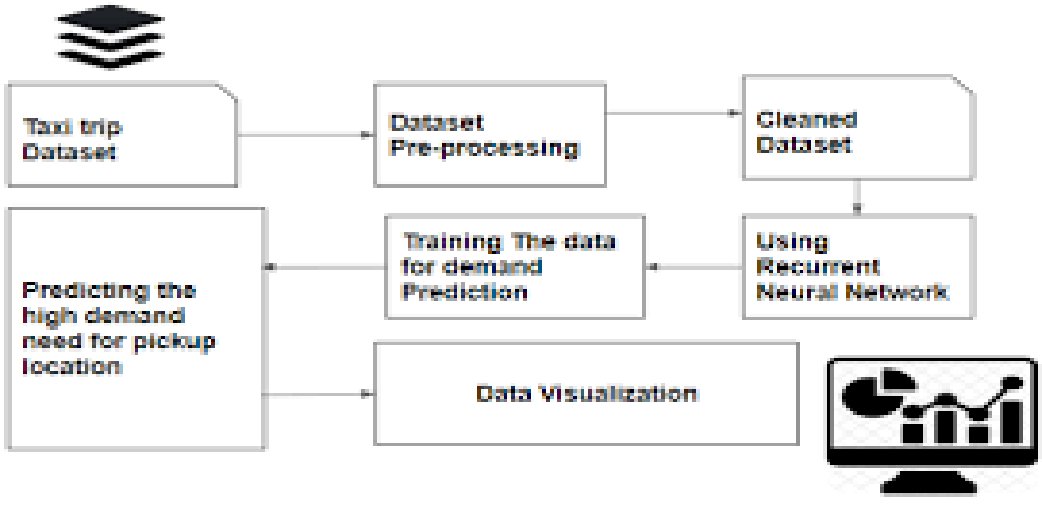

Figure 4.3: Uber Fare Prediction Model

In the above figure shows how the uber fare prediction model is going to work with the help of the dataset.

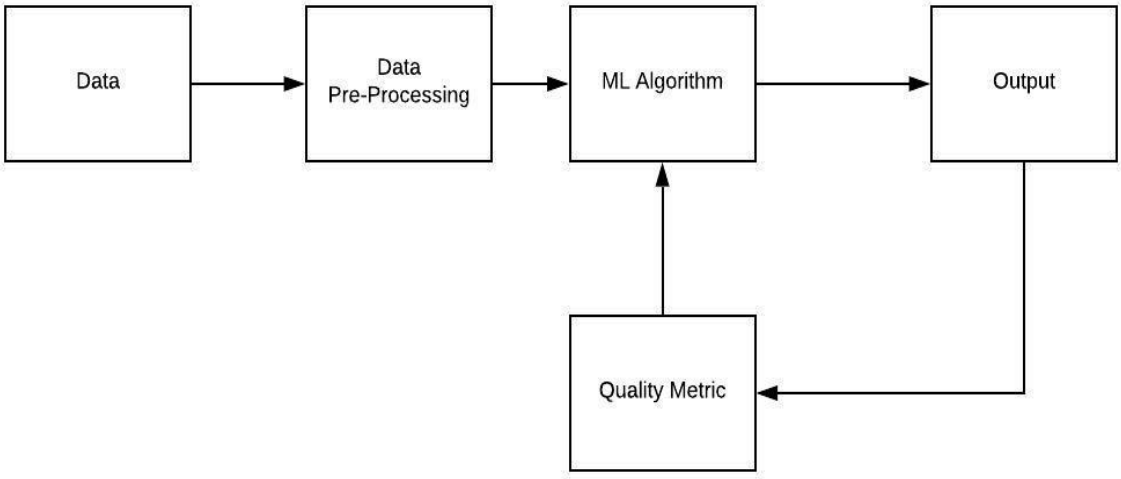

Figure 4.4: Flowchart of Linear Regression Model

The above figure is the flowchart of Random Forest Regression Model. It will helps us to understand the flow of regression.

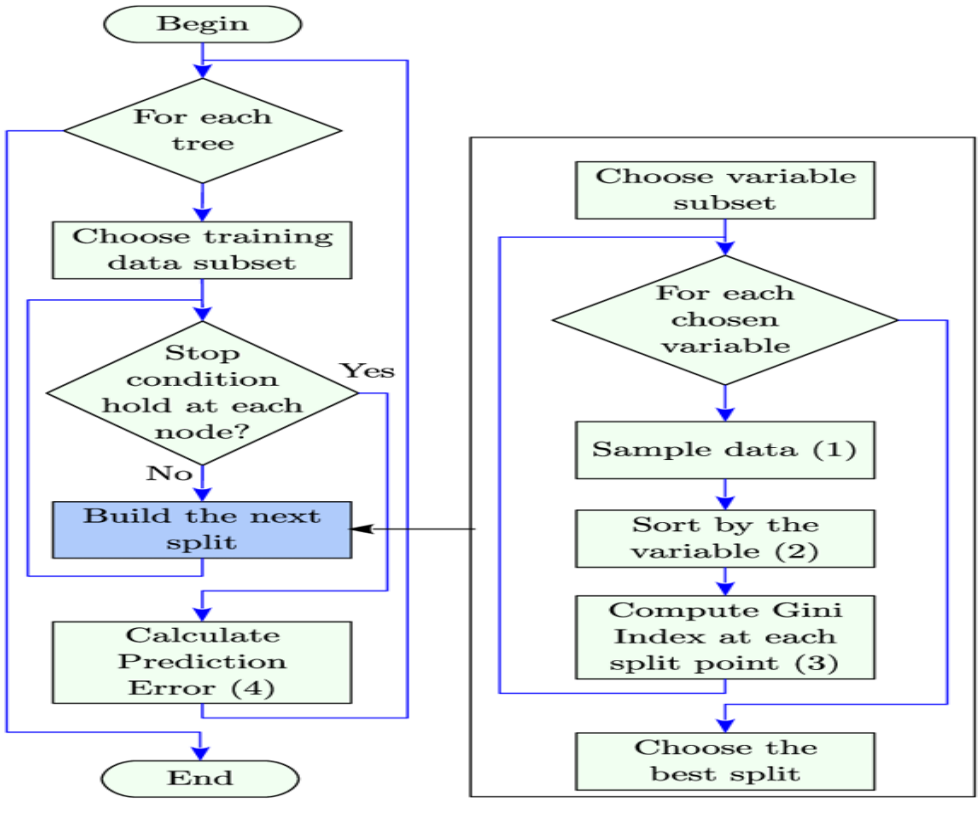

Figure 4.5: Flowchart of Random Forest Regression Model

The above figure is the flowchart of Random Forest Regression Model. It will helps us to understand the flow of regression.

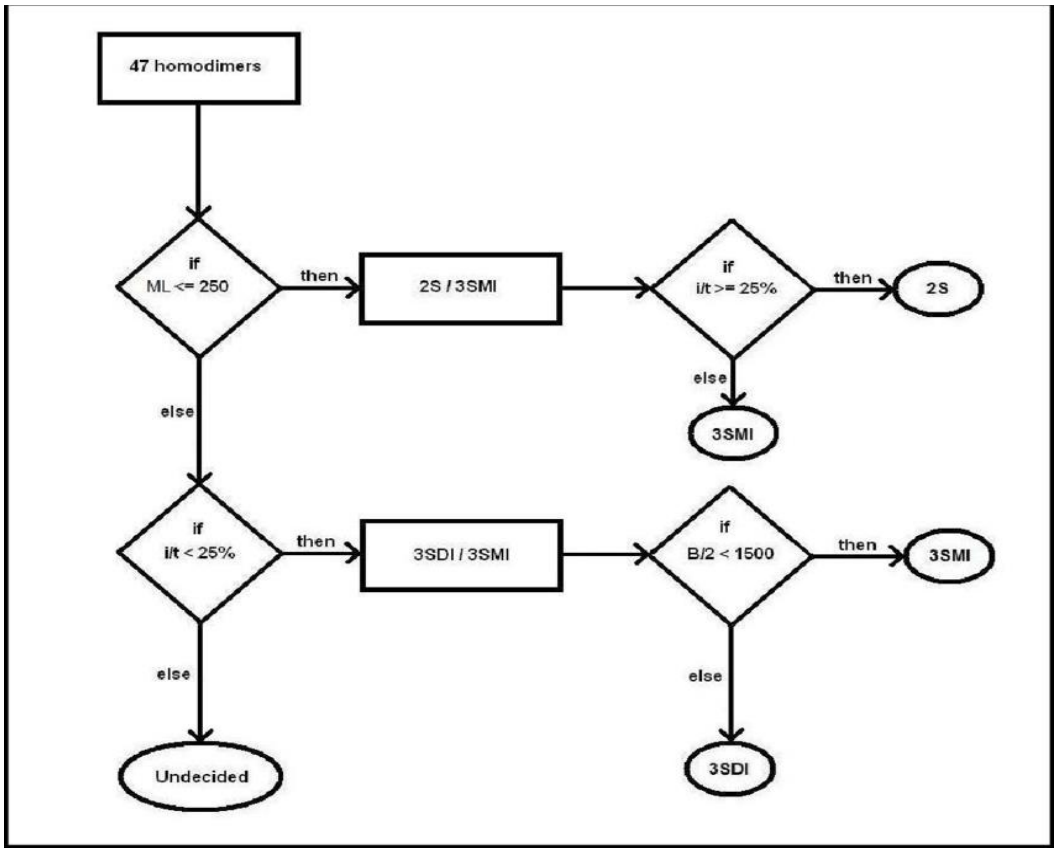

Figure 4.6: Flowchart of Decision Tree Model

The above figure is a flowchart of Decision Tree Model Regression. It helps in understanding the flow of the decision tree

4.2 ALGORITHMS

a) **Linear Regression:** Linear regression algorithms show a linear relationship between a dependent (y) and one or more independent (y) variables, hence called linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable. When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error. The different values for weights or the coefficient of lines (a0, a1) gives a different line of regression, so we need to calculate the best values for a0 and a1 to find the best fit line, so to calculate this we use cost function. As a baseline prediction, the mean duration and fare from the training set were used to predict a constant value for the validation set. From feature selection, the linear regression with all covariates available at the pickup time was predicted to be the best for both duration

and fare prediction. The linear regression model finds the set of θ coefficients that minimize the sum of squared errors.

```
SUB Regress(x, y, n, a1, a0, syx, r2)

  sumx = 0: sumxy = 0: st = 0
  sumy = 0: sumx2 = 0: sr = 0
  DOFOR i = 1, n
    sumx = sumx + x_i
    sumy = sumy + y_i
    sumxy = sumxy + x_i*y_i
    sumx2 = sumx2 + x_i*x_i
  END DO
  xm = sumx/n
  ym = sumy/n
  a1 = (n*sumxy - sumx*sumy)/(n*sumx2 - sumx*sumx)
  a0 = ym - a1*xm
  DOFOR i = 1, n
    st = st + (y_i - ym)^2
    sr = sr + (y_i - a1*x_i - a0)^2
  END DO
  syx = (sr/(n - 2))^0.5
  r2 = (st - sr)/st

END Regress
```

Figure 4.7: Algorithm of Linear Regression

b) **Decision Tree Regression:** Decision Tree is one of the most commonly used, practical approaches for supervised learning. It can be used to solve both Regression and Classification tasks with the latter being put more into practical application. It is a tree-structured classifier with three types of nodes. The Root Node is the initial node which represents the entire sample and may get split further into further nodes. The *Interior No*des represent the features of a data set and the branches represent the decision rules. Finally, the *Leaf Nodes* represent the outcome. This algorithm is very useful for solving decision-related problems. The dataset is broken into smaller subsets with leaf and decision nodes. Greedy search method is employed by the decision tree and the ID3 algorithm is used in decision tree regression model and the results are predicted based on standard deviation reduction, the same set of rules is adopted in the model for taxi price prediction.

GenDecTree(Sample S, Features F)

Steps:

1. ***If*** *stopping_condition(S, F) = true* ***then***
   a. *Leaf = createNode()*
   b. *leafLabel = classify(s)*
   c. ***return*** *leaf*
2. *root = createNode()*
3. *root.test_condition =findBestSpilt(S,F)*
4. *V = {v | v a possible outcomecfroot.test_condition}*
5. ***For each*** *value v ∈ V:*
   a. S_v *= {s | root.test_condition(s) = v and s ∈ S };*
   b. *Child = TreeGrowth (*S_v*,*F*);*
   c. *Add child as descent of root and label the edge {root → child} as v*
6. ***return*** *root*

Figure 4.8: Decision Tree Regression Algorithm

c) **Random Forest Regression: Random Forest Regression** is a supervised learning algorithm that uses **ensemble learning** method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. The model with the least error and highest variance is considered as the best model. Therefore, the Random Forest model is finalized for deployment as it has the least RMSE values compared to other models and highest variance score as shown in table 3 inferring there is 91% of variability in the dependent variable 'fare amount' caused by the independent variables.

Algorithm 1: Pseudo code for the random forest algorithm

To generate c classifiers:
for i = 1 to c **do**
  Randomly sample the training data D with replacement to produce D_i
  Create a root node, N_i containing D_i
  Call BuildTree(N_i)
end for

BuildTree(N):
if N contains instances of only one class **then**
  return
else
  Randomly select x% of the possible splitting features in N
  Select the feature F with the highest information gain to split on
  Create f child nodes of N, N_1,..., N_f, where F has f possible values (F_1, ... , F_f)
  for i = 1 to f **do**
    Set the contents of N_i to D_i, where D_i is all instances in N that match
    F_i
    Call BuildTree(N_i)
  end for
end if

Figure 4.9: Random Forest Regression Algorithm

Chapter 5

IMPLEMENTATION

5.1 PSEUDOCODE

a) Exploratory Data Analysis:

```
[ ] import numpy as np # linear algebra
    import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

```
[ ] train_df=pd.read_csv(r"/content/drive/MyDrive/train.csv",nrows=10000)
    #test_df=pd.read_csv(r"/content/drive/MyDrive/Internship/test.csv")
    print (train_df.shape)
    print (train_df.columns)
    #print (test_df.shape)
    #print (test_df.columns)
```

```
(10000, 8)
Index(['key', 'fare_amount', 'pickup_datetime', 'pickup_longitude',
       'pickup_latitude', 'dropoff_longitude', 'dropoff_latitude',
       'passenger_count'],
      dtype='object')
```

Figure 5.1: Import Dataset

In the above figure, we are importing numpy, pandas, storing training dataset by calling read_csv and printing the shape and the columns of dataset.

b) Data Description:

```
[ ] train_df.describe()
```

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
count	9718.000000	9718.000000	9718.000000	9718.000000	9718.000000	9718.000000
mean	11.227631	-73.839094	40.676157	-73.854353	40.648201	1.647767
std	9.324101	3.661502	4.902851	3.488590	3.212462	1.273914
min	-2.900000	-74.438233	-74.006893	-74.185996	-73.994392	0.000000
25%	6.000000	-73.992255	40.736006	-73.991290	40.736635	1.000000
50%	8.500000	-73.982090	40.753391	-73.980501	40.754476	1.000000
75%	12.500000	-73.968285	40.768042	-73.965388	40.768596	2.000000
max	165.000000	40.766125	401.083332	40.802437	41.366138	6.000000

Figure 5.2: Dataset Description

In the above figure, we can see the description of training dataset.

c) Calculating missing values:

```
missing_val = pd.DataFrame(train_df.isnull().sum())
missing_val = missing_val.reset_index()
#Rename variable
missing_val = missing_val.rename(columns = {'index': 'Variables',
0: 'count'})
missing_val['Missing_percentage'] = (missing_val['count']/len(tra
in_df)*100)
#sort in descending order
missing_val = missing_val.sort_values('Missing_percentage', ascen
ding = False).reset_index(drop = True)
print(missing_val)
```

d) Outliner Analysis:

- Irregular fare amount converted to N/A

```
# irregular fare_amount are converted to NA

train_df.loc[train_df['fare_amount']<0 , 'fare_amount']=np.nan
train_df.loc[train_df['fare_amount'] > 30, 'fare_amount']=np.nan
train_df=train_df.dropna()
```

```
[ ] train_df.shape

(9189, 8)
```

Figure 5.3: Irregular Fare Converted to N/A

In the above figure, we are dropping the rows which is having irregular fare amount by converting them to NA.

- Calculate distance and set boundary

```
[ ] #save numeric data names
    coutliers = [ 'pickup_longitude', 'pickup_latitude', 'dropoff_longitude', 'dropoff_latitude']
    for list in coutliers:
        #Detect and replace with NA
        #Extract quartiles
        q75, q25 = np.percentile(train_df[list], [75 ,25])

        #Calculate IQR
        iqr = q75 - q25

        # #Calculate inner and outer fence
        minimum = q25 - (iqr*1.5)
        maximum = q75 + (iqr*1.5)

        # #Replace with NA
        train_df.loc[train_df[list] < minimum,list] = np.nan
        train_df.loc[train_df[list] > maximum,list] = np.nan

        # #Calculate missing value
        missing_val = pd.DataFrame(train_df.isnull().sum())
```

Figure 5.4: Distance and Boundary Calculation

In the above figure, we calculate the distance to be travelled by the customer and replacing to NA as the distance cannot be travelling.

- Input missing values with outlier mean value

```
#As Mean is the best method, we impute missing values/ in this case outlier values with mean

train_df['pickup_longitude'] = train_df['pickup_longitude'].fillna(train_df['pickup_longitude'].mean())
train_df['pickup_latitude'] = train_df['pickup_latitude'].fillna(train_df['pickup_latitude'].mean())
train_df['dropoff_longitude'] = train_df['dropoff_longitude'].fillna(train_df['dropoff_longitude'].mean())
train_df['dropoff_latitude'] = train_df['dropoff_latitude'].fillna(train_df['dropoff_latitude'].mean())

#imputed with mode for categorical variables
train_df['passenger_count'] = train_df['passenger_count'].fillna(int(train_df['passenger_count'].mode()))
```

Figure 5.5: Outline Missing Values with mean

e) Parametric Data Analysis:

After identifying the approach, the next step is pre-processing the data. Looking at data refers to exploring the data, refining the data as well as visualizing the data through graphs and plots. This is often called as Exploratory Data Analysis (EDA). To initiate this process, any of the probability distributions of the variables are considered. Most analysis like regression, require the data to be normally distributed. This can be

visualized at a glance by looking at the probability distributions or probability density functions of the variable

- Density vs Fare_amount:

```
sns.distplot(train_df['fare_amount'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distril
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f7ff4ddd
```

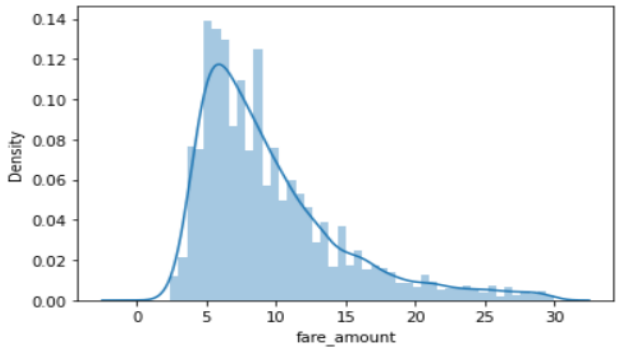

Figure 5.6(a): Density vs Fare Graph

- Density vs Pickup_latitude:

```
sns.distplot(train_df['pickup_latitude'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/dis
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f7ff4
```

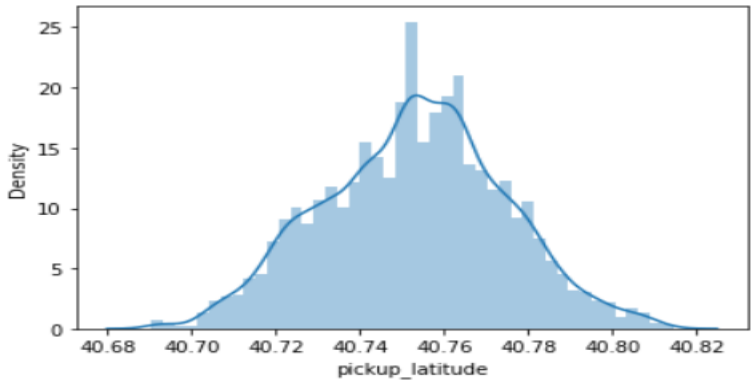

Figure 5.6(b): Density vs Pickup_latitude Graph

- Density vs Pickup_longitude:

```
sns.distplot(train_df['pickup_longitude'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distr
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f7ff27
```

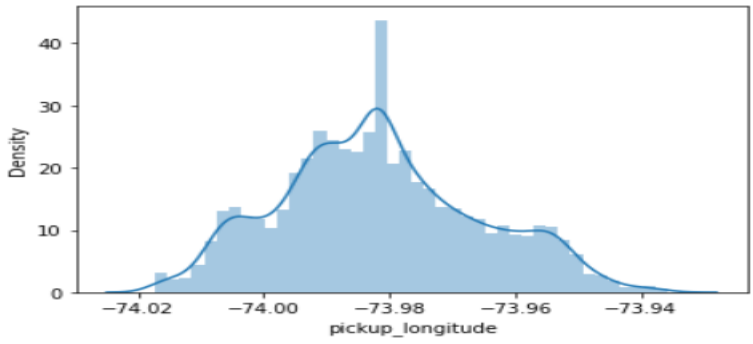

Figure 5.6(c): Density vs Pickup_longitude Graph

- Density vs Dropoff_Latitude:

```
sns.distplot(train_df['dropoff_latitude'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/dis
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f7ff2
```

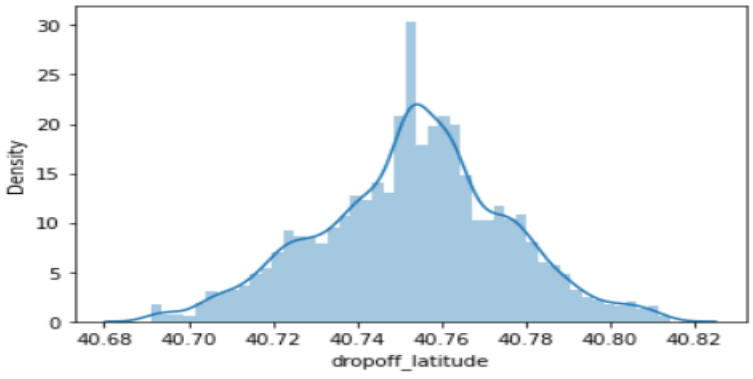

Figure 5.6(d): Density vs Dropoff_latitude Graph

- Density vs Dropoff_longitude:

```
sns.distplot(train_df['pickup_longitude'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distri
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f7ff27c6
```

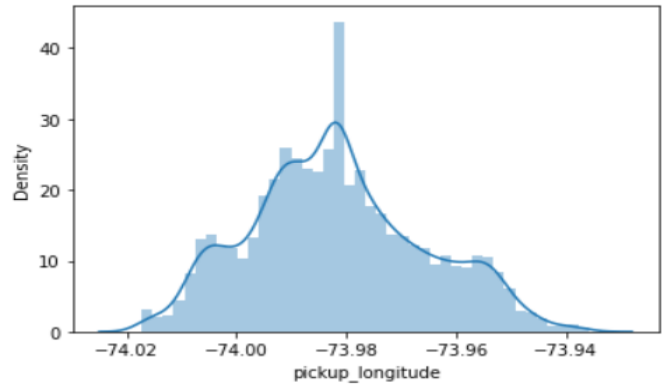

Figure 5.6(e): Density vs Dropoff_longitude

f) Calendar to calculate surge of special days:

```
[ ] import calendar
    train_df['day']=train_df['pickup_datetime'].apply(lambda x:x.day)
    train_df['hour']=train_df['pickup_datetime'].apply(lambda x:x.hour)
    train_df['weekday']=train_df['pickup_datetime'].apply(lambda x:calendar.day_name[x.weekday()])
    train_df['month']=train_df['pickup_datetime'].apply(lambda x:x.month)
    train_df['year']=train_df['pickup_datetime'].apply(lambda x:x.year)

[ ] train_df.head()
```

	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	day	hour	weekday	month	year
0	2009-06-15 17:26:21.0000001	4.5	2009-06-15 17:26:21+00:00	-73.981706	40.721319	-73.980109	40.712278	1.0	15	17	Monday	6	2009
1	2010-01-05 16:52:16.0000002	16.9	2010-01-05 16:52:16+00:00	-74.016048	40.711303	-73.979268	40.782004	1.0	5	16	Tuesday	1	2010
2	2011-08-18 00:35:00.00000049	5.7	2011-08-18 00:35:00+00:00	-73.982738	40.761270	-73.991242	40.750562	2.0	18	0	Thursday	8	2011
3	2012-04-21 04:30:42.0000001	7.7	2012-04-21 04:30:42+00:00	-73.987130	40.733143	-73.991567	40.758092	1.0	21	4	Saturday	4	2012
4	2010-03-09 07:51:00.000000135	5.3	2010-03-09 07:51:00+00:00	-73.968095	40.768008	-73.956655	40.783762	1.0	9	7	Tuesday	3	2010

Figure 5.7: Calendar

In the above figure, we are creating a new columns and storing the day, month, year, hour, weekdays which will be useful for calculating cost of the ride.

g) Linear Corelation:

```
def corr_heatmap(df):
    corr_data = df.corr()
    fig, ax = plt.subplots(figsize=(10,8))
    # Add title
    # Heatmap showing the amount of genomes with the same MIC for
 each MIC, by antibiotic
    sns.heatmap(corr_data, annot=corr_data, cmap='Blues', cbar=Tr
ue, fmt='.2f')
    # Rotate the tick labels and set their alignment.
    plt.setp(ax.get_xticklabels(), rotation=45, ha='right', rotat
ion_mode='anchor')
    plt.show()
corr_heatmap(train_df)
```

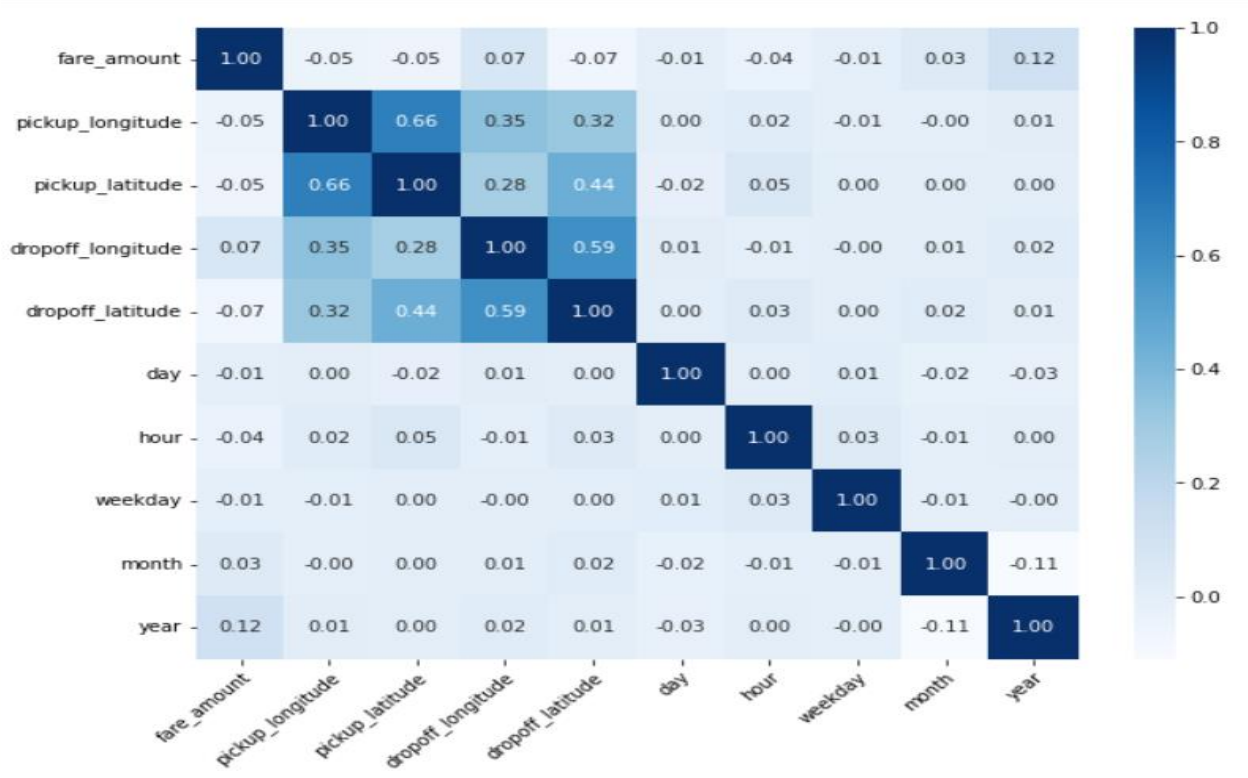

Figure 5.8: Linear Correlation Heat Map

The above figure shows the linear Correlation between the parameters using Heat Map. Linear Correlation is a measure of the degree of association between variables.

$$R = \frac{n(\sum xy)-(\sum x)(\sum y)}{\sqrt{[n\sum x^2-(\sum x)^2]\,[n\sum y^2-(\sum y)^2]}}$$

Where R is Linear Correlation Coefficient.

If R is 1.0, it is said to be positive correlation between the two variables.
If R is 0.4, it is said to be no correlation between the two variables.
If R is 0.0, it is said to be negative correlation between the two variables.

Chapter 6

RESULTS

6.1 RESULTS SNAPSHOTS

```
dtr_rmse=np.sqrt(mean_squared_error(predictions_DT, y_test))
print("RMSE value for decision tree regression is ",dtr_rmse)
```

```
RMSE value for decision tree regression is  4.700253355984775
```

Figure 6.1: RMSE value for Decision Tree

The above figure shows the Random Mean Square Error value for Decision Tree Regression.

```
#lets calculate rmse for linear Regression model
from sklearn.metrics import mean_squared_error
lrmodelrmse = np.sqrt(mean_squared_error(predictedvalues, y_test))
print("RMSE value for Linear regression is", lrmodelrmse)
```

```
RMSE value for Linear regression is 5.029844743227887
```

Figure 6.2: RMSE value of Linear Regression

The above figure shows the Random Mean Square Error value for Linear Regression.

```
rfrmodel_rmse=np.sqrt(mean_squared_error(rfrmodel_pred, y_test))
print("RMSE value for Random forest regression is ",rfrmodel_rmse)
```

```
RMSE value for Random forest regression is  3.049356294644267
```

Figure 6.3: RMSE value of Random Forest Regression

The above figure shows the Random Mean Square Error value for Random Forest Regression.

By comparing all 3 Regression model, Random forest regression has lowerst RMSE value. So, we will use Random Forest Regression model to predict the cost for travelling in Uber.

```
[ ] test_df.head()
```

	key	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	dist	fare_amount
0	2015-01-27 13:08:24.0000002	2015-01-27 13:08:24 UTC	-73.973320	40.763805	-73.981430	40.743835	1	2.323260	7.600
1	2015-01-27 13:08:24.0000003	2015-01-27 13:08:24 UTC	-73.986862	40.719383	-73.998886	40.739201	1	2.425353	8.770
2	2011-10-08 11:53:44.0000002	2011-10-08 11:53:44 UTC	-73.982524	40.751260	-73.979654	40.746139	1	0.618628	5.420
3	2012-12-01 21:12:12.0000002	2012-12-01 21:12:12 UTC	-73.981160	40.767807	-73.990448	40.751635	1	1.961033	7.990
4	2012-12-01 21:12:12.0000003	2012-12-01 21:12:12 UTC	-73.966046	40.789775	-73.988565	40.744427	1	5.387301	13.007

Figure 6.4: Predicted Fare

The above figure shows the Predicted Cost of the testing dataset using Random Forest Regression.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION

- The quality of a regression model depends on the matchup of predictions against actual values. In regression problems, the dependent variable is continuous. In classification problems, the dependent variable is categorical.
- Random Forest can be used to solve both regression and classification problems. Decision trees are nonlinear; unlike linear regression, there is no equation to express the relationship between independent and dependent variables.
- Out of the three models left, Random Forest is the best model as it has the lowest RMSE score, which explains the highest variability and tells us how well the model fits in this data.

FUTURE ENHANCEMENTS

- Forecasting is critical for building better products, improving user experiences, and ensuring the future success.
- We can use GPS to find the correct location of uber and user.
- We propose to develop a mobile Application that would essentially use insights from our model to position taxicabs with more efficiency.
- Our model could be used by both taxi driver as well as customer.

REFERENCES

[1] John D. Kelleher, Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies (The MIT Press), 1 st Edn.

[2] Data science work with public datasets, [Online] Available: https://www.kaggle.com, [3] Chong Ho Yu, Exploratory data analysis in the context of data mining and re-sampling, International Journal of psychological research, 2010, vol.3, No.1

[4] Ruben Oliva Ramos, Jen Stirrup, Advanced Analytics with R and Tableau, Packt Publishing, 2017, ISBN: 9781786460110

[5] Machine Learning in Python,[Online] Available https://scikit-learn.org/stable/

[6] T. Divya and A. Sonali, "A survey on Data Mining approaches for Healthcare", International Journal of Biosciences and Bio-Technology, vol. 5, no. 5, (2013), pp. 241-266.

[7] Sebastian Raschka, Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning, [Online] Available https://arxiv.org/abs/1811.12808,2016

[8] Weijie Wang and Yanmin Lu, Analysis of the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE) in Assessing Rounding Model, ICMEMSCE, IOP Publishing, 324 (2018), doi:10.1088/1757-899X/324/1/012049